Carbon Paper: Peering into Turla’s second stage backdoor

By ESET Research

The following analysis is based on the version 4.x of msximl. This component may have changed in the latest versions.

Configuration fetching

Besides the code in the "Configuration fetching" thread from the orchestrator (which is similar), a field "sethttp1" is retrieved from the [TRANSPORT] section.

If this value is set, HTTP 1.1 will be used for future connections.

Tasks execution

The tasks are retrieved from the C&C server.

The tasks to be executed by the communication library are listed in the file "b9s3coff.ax" (for Carbon v3.7x) or "cifrado.xml" (for Carbon v3.8x).

Each line of this file is composed in the following way:

- task_id | task_filepath | task_config_filepath | task_result_filepath | task_log_filepath

The task file and its config are decrypted (CAST-128) and the task executed. There are some options that are retrieved from the Carbon configuration file: "time2task" is used to set a timeout for the task execution (1 hour by default) while "task_min" and "task_max" are used as to wait a random time between the execution of the tasks on the task list (the random time will be set between "task_min" and "task_max"). These parameters come from the section [TIME].

If the task is a valid DLL file, it is loaded in the current process memory and a new thread is created to execute its "start" export. Otherwise, this is probably a command to execute. In this case, the configuration file is parsed. Like the Carbon configuration file, the task configuration file is formed as a windows "inf" file and should contain a section [CONFIG] with the following parameters:

- [CONFIG]
    - NAME ("cmd.exe" by default)
    - ARG
    - RESULT ("stdout" by default)
    - COMPRESSION ("yes" by default)
    - DELETE ("no" by default)

The command with its arguments is executed through CreateProcess() function and the results are stored in the %task_result_filepath% file if the RESULT option is not set to "stdout", while error or additional info are added to the task logfile.

If the option RESULT is set to stdout:

- the result task file is deleted and the task execution output is written to the log task file
- the line "task_id | "1" | task_log_filepath | object_id" is added at the end of the file "C_56743.NLS" (Carbon 3.7x) or "dsntype.gif" (Carbon 3.8x)

Otherwise:

- the task execution is outputted to the task result file and the errors / additional info to the task logfile
- the line "task_id | "2" | task_log_filepath | task_result_filepath | object_id" is added at the end of the file "C_56743.NLS" (Carbon 3.7x) or "dsntype.gif" (Carbon 3.8x)

In both cases, the same line as written into "C_56743.NLS" is also written into the field "run_task" from the Carbon configuration file. It might be intended as a way for the malware authors to know which is the last task executed when the configuration file is sent to the C&C server (check "Configuration file backup").

Log rotation

The logfile is backed up and sent to the C&C server periodically (by default this is done every two hours).

Like the backup of the configuration file, this action is performed only during specific times of the day.
Indeed, the log rotation will be performed only when the current local time is in the range of what is defined in the configuration file.

The fields "wh_min" and "wh_max" are retrieved from the [TIME] section of the configuration file (by default: 8 and 20). The thread will execute the code only if the current hour of the day is between %wh_min% and %wh_max%.

If there is a value for the attribute "timestop" from the [TIME] section (which looks like "wDay:wMonth:wYear:wHour:wMinute"), the thread will execute the code only after this specific datetime.

The attributes "lastsend" and "logperiod" from the [LOG] section are used to specify a delay time when to backup and send the log to the C&C server. (by default: every two hours).

A temporary file with a random name prefixed by "~D" is created in the folder "208" (for Carbon v3.7x) or "1033" (for Carbon v3.8x). The logfile content is copied into this new file. It is then compressed with Bzip2 and encrypted (CAST-128).

A new line is added at the end of the file "C_56743.NLS" (for Carbon v3.7x) or "dsntype.gif" (for Carbon v3.8x):

- "10|1|%s|%s"
    - 1st field: an ID to identify the file as a logfile
    - 2nd field: 1 (file to be sent to the C&C server)

- 3rd field: the temp file path
- 4rd field: the victim uuid

Last but not least, the attribute "lastsend" is updated with the current time and the original logfile is deleted.

Communication with the C&C server

The code of this thread is used to retrieve new tasks from the C&C server, to send new files to the server (the files listed in the file "C_56743.NLS" / "dsntype.gif") and to send the new tasks to the orchestrator.

First request

A random C&C server address is chosen from the ones in the section "CW_INET". If the port and HTTP resource path are not specified, the default is to use port 80 and "/javascript/view.php".

A user agent is set up in the following way:

- the version of Internet Explorer is retrieved through the registry key: "HKLM\Software\Microsoft\Internet Explorer\Version" and is concatenated to the string "Mozilla/4.0 (compatible; MSIE %d.0; "
  - example: "Mozilla/4.0 (compatible; MSIE 8.0.6001.18702.0;"
- concatenate the previous string with the OS major/minor version values (through GetVersionExA())
  - "Mozilla/4.0 (compatible; MSIE 8.0.6001.18702.0; Windows NT 5.1; Trident/4.0"
- enumerate the values key in "HKLM\Software\Microsoft\Windows\CurrentVersion\Internet Settings\5.0\User Agent\Post Platform" and concatenate each value to the previous string and then append a closing paren.
  - example: "Mozilla/4.0 (compatible; MSIE 8.0.6001.18702.0; Windows NT 5.1; Trident/4.0; .NET CLR 2.0.50727; .NET CLR 3.0.30729; .NET CLR 3.5.30729; .NET4.0C; .NET4.0E; Media Center PC 6.0; SLCC2)

The field "trans_timemax" from the section [TIME] is retrieved. It is used to set the timeout for internet requests (through InternetSetOption()). It has a value of 10 minutes by default.

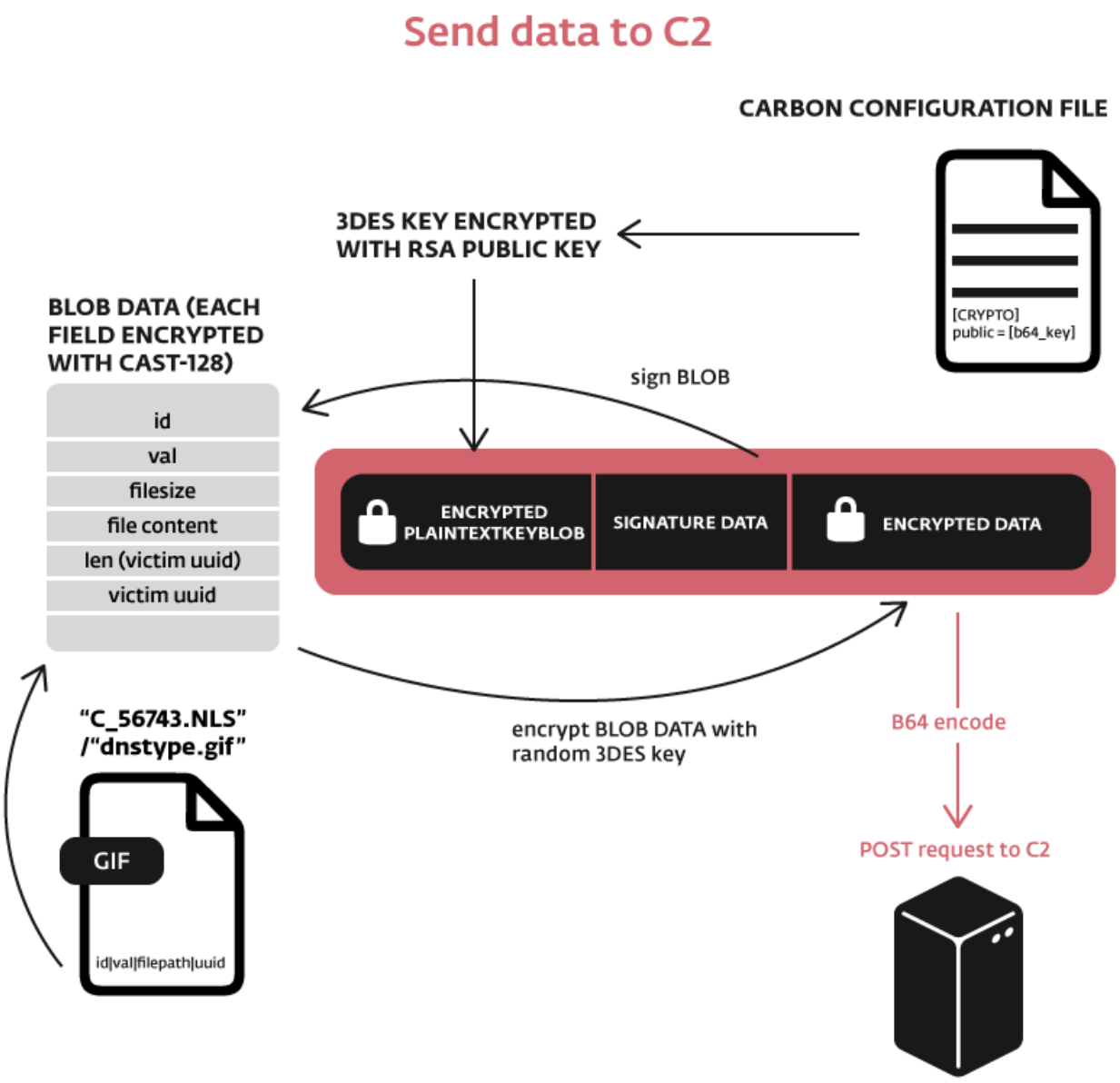
A first GET request is performed on the root page of the C&C web server to check that the host is alive. If no packet capture is running on the system, a new request is done on the C&C server to check if new tasks are available. A "PHPSESSID" cookie is added to the request with the victim uuid as its value. A header "Referer" is added as well and set to the C&C server URL.

The malware is expecting to get an answer to the GET request similar to:

- <input name="%name%" value="%data_in_b64%">

If the field "value" contains something, a new task is available.

Send data to the server



If the file "C_56743.NLS" / "dsntype.gif" is not empty, it means there are data to be sent the C&C server. The file is parsed and the last line is retrieved. It contains details about the data to be sent. A data blob is built and each of the following fields is encrypted with CAST-128:

- id | val | tmp_filesize | tmp_content | [OPTIONAL (if val == 2) tmp2_filesize | tmp2_content] | len_object_id | object_id
    - id = the type of data to send to the C&C server, it can be:
        - 10: log backup
        - 11: configuration file
        - 20: a cryptographic key
        - otherwise: an id associated to a task, it can be the result of a task or an error log in the case of task execution failure
    - val = 1 if there is only one file to send, 2 if there are two files

- object_id = the victim uuid

If the field "dtc" from the section [CRYPTO] of the configuration file is set to 0, this whole blob is base64 encoded and sent to the C&C server through a POST request.

Otherwise, another layer of encryption is used. In this case, the data blob is signed and a random 3DES key is used to encrypt it. Because the 3DES key is randomly generated and the server needs it to decrypt the data, the key is encrypted with the server public key. The server key is retrieved from the field "publicc" of the section [CRYPTO] from the configuration file.

This new blob (encrypted_key | signature_data | encrypted data) is encoded in base64 and sent to the C&C server through a POST request.

In order to avoid detection based on the data size sent in a request, the blob can be fragmented into several packets. An option in the configuration file ("post_frag" in the section [TRANSPORT]) defines whether the blob will be fragmented or sent in only one POST request.

If this option is set to "yes", the blob is divided into several fragments of a specific size. This size comes from another field in the configuration file: "post_frag_size".

An additional header will be added to the request:

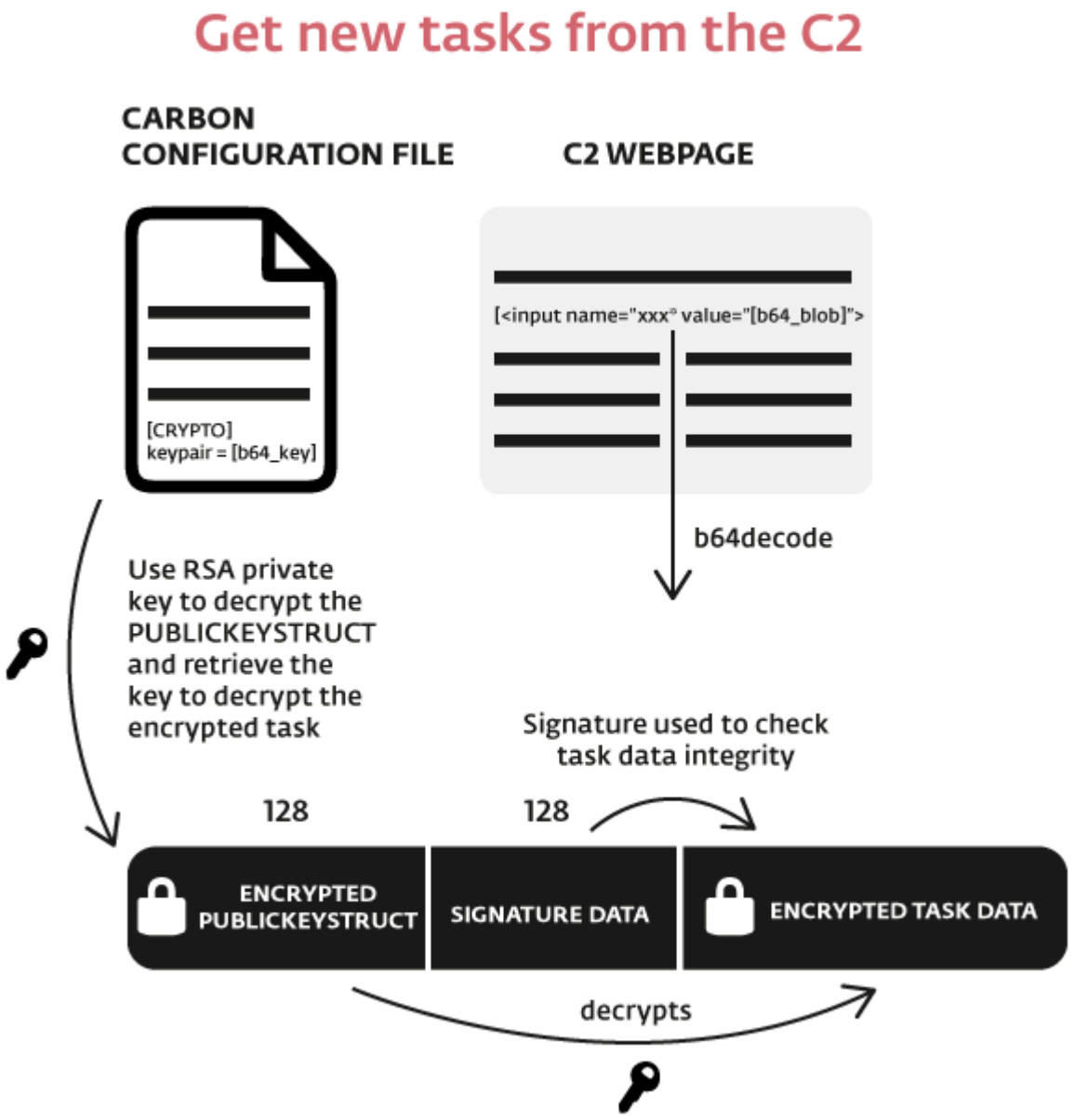
- "Content-Range: bytes %u-%u/%u; id=%u\r\n", i, i+(fragment_size-1), data_size, task_id"

If the option http11 is set, a specific header is added as well:

- "Expect: 100-continue\r\n"

For each fragments sent, the fields "post_frag_size" and "pfslastset" from the config file (section [CW_INET_RESULTS]) are updated with the fragment size and the timestamp.

Get new tasks



New tasks are retrieved from the C&C server by parsing the html page. The malware expects to find the html tag <input> in the page with a base64 encoded blob in its "value" attribute. Once decoded, this blob contains:

- an encrypted block of 128 bytes that contains a structure "PUBLICKEYSTRUC" followed by a cryptographic key (probably a 3DES key)
- signature data (128 bytes) to verify the integrity of the next block
- a block of encrypted data that contains the task

The malware uses an RSA private key (retrieved from the field "keypair" from the section [CRYPTO] of the configuration file) to decrypt the first block and then uses the freshly decrypted key to decrypt the third block. This block of data can be either:

- a task to be executed [6]
    - the data are decrypted and stored in few temporary files, the task (a command or a PE file) and its configuration file in the "Nls" folder and the output files (file that will contains the task results and

the logfile) are stored in the folder "0208" (or respectively "0409" and "1033" for Carbon v3.8x)
    - task_id | task_filepath | task_config_filepath | task_result_filepath | task_log_filepath
    - this line is appended to beginning of the file "b9s3coff.ax" (cifrado.xml on v3.8x)
- a task to be executed by the orchestrator [7]
    - the data are decrypted and stored in few temporary files (the task, its configuration etc) in the "Nls" and "0208" folder (or "0409" and "1033" for Carbon v3.8x)
    - depending of the content of the data, one of these lines will be added to the beginning of the file "a67ncode.ax" (encodebase.info on v3.8x)
        - task_id | task_filepath | task_config_filepath | task_result_filepath | task_log_filepath
        - task_id | task_filepath | task_config_filepath | task_result_filepath | task_log_filepath | execution_mode | username | password
        - task_id | task_filepath | task_config_filepath | task_result_filepath | task_log_filepath | execution_mode
- a new RSA server public key
    - in this case, the configuration file is updated with the new key encoded in base64 (field publicc)
- data to be sent to an instance of Carbon running in another computer in the same network
    - the data can contains a specific IP address and port, a named pipe or a named pipe with a username and password.

Check Internet availability

Each hour, the internet connection is checked. A first check is done by calling the function InternetAttemptConnect(). If it works, another test is done by sending HTTP GET requests to the following websites:

- www.google.com
- www.yahoo.com
- www.bing.com
- update.microsoft.com
- windowsupdate.microsoft.com
- microsoft.com

An event is used to notify the other threads in case of the loss of Internet access.

Configuration file backup

Similar to the logfile, the configuration file is also periodically backed up and sent to the C&C server. The thread executes the code in a specific range of time (between 8h and 20h by default) [8].

The value "configlastsend" is retrieved from the section [TIME] of the configuration file. If the config file has been sent over a month ago, the config file is copied into a temporary file with a random name prefixed by "~D" in the folder "208" (for Carbon v3.7x) or "1033" (for Carbon v3.8x). This file is then encrypted with CAST-128 algorithm.

To notify the thread that communicates with the C&C server that a new file is ready to be sent to the server, the following line is appending to the file "C_56743.NLS" (for Carbon v3.7x) or "dsntype.gif" (for Carbon v3.8x):

- "11|1|%s|%s"
    - 1st field: an ID to identify the file as a config file
    - 2nd field: 1 (file to be sent to the C&C server)
    - 3rd field: the temp filepath
    - 4rd field: the victim uuid

Last but not least, the attribute "configlastsend" is updated with the current time.

Source: https://www.welivesecurity.com/2017/03/30/carbon-paper-peering-turlas-second-stage-backdoor/